

1. Longest common subsequence

Suppose we are given two arrays, A and B , of n and m positive numbers (or two strings of n and m characters)

$$A = [a_1, a_2, \dots, a_n],$$

$$B = [b_1, b_2, \dots, b_m].$$

A **common subsequence** of A and B is a subsequence $[a_{i_1}, a_{i_2}, \dots, a_{i_k}]$ of elements from A and a subsequence $[b_{j_1}, b_{j_2}, \dots, b_{j_k}]$ of elements from B such that $a_{i_\nu} = b_{j_\nu}$ for each $1 \leq \nu \leq k$. Notice that the elements in the common subsequence need not be contiguous in A or B .

Given a common subsequence of A and B , we can compute its length. The “longest common subsequence” problem is to find the maximum possible length over all possible common subsequences from the given arrays (or strings) A and B .

Let us consider how to solve this problem using dynamic programming. We need to come up with a definition of a subproblem for which we can write a recurrence relation relating the solution of the subproblem to the solutions of other (smaller) subproblems. The most obvious choice for a subproblem is

$$L(i, j) = \text{length of the longest subsequence common to } [a_1, \dots, a_i] \text{ and } [b_1, \dots, b_j].$$

As we shall now show, this obvious choice of subproblem is a good choice. We can write a recurrence relation relating subproblem $L(i, j)$ to its (smaller) subproblems $L(r, s)$ with $r < i$ or $s < j$.

The idea behind the recurrence relation is this. Given indices i and j , consider elements a_i from A and b_j from B . We have two mutually exclusive cases, either $a_i = b_j$ or $a_i \neq b_j$. Let us consider these two cases one at a time.

Suppose that we have $a_i = b_j$. Then the element a_i (and also b_j) *must* be the tail end of the common subsequence that solves the subproblem $L(i, j)$ (why?). What about the rest of this common subsequence? It must be $L(i-1, j-1)$. That is, the common subsequence that solves $L(i, j)$ must have a_i tacked on to the end of the common subsequence that solves $L(i-1, j-1)$. So in this case we have

$$L(i, j) = 1 + L(i-1, j-1) \quad \text{if } a_i = b_j.$$

That is what our recurrence relation looks like in this case.

Now suppose that we are in the other case, where $a_i \neq b_j$. In this case we can say for sure that we cannot have *both* a_i and b_j as part of the solution to $L(i, j)$ (if both a_i and b_j were part of the solution to $L(i, j)$, then, since a_i is the last item we can choose from A and b_j is the last item we can choose from B , it must be that a_i and b_j are the last item in the common subsequence, but that implies $a_i = b_j$, a contradiction). If we cannot have both a_i and b_j as part of the solution to $L(i, j)$, that gives us three (mutually exclusive) cases. Either a_i is part of the solution to $L(i, j)$ and b_j is not, or b_j

is part of the solution to $L(i, j)$ and a_i is not, or neither a_i nor b_j is part of the solution to $L(i, j)$. Let us look at these three cases one at a time.

Suppose that $a_i \neq b_j$, and a_i is not part of the solution to $L(i, j)$ but b_j is part of this solution. Since we know that $L(i, j)$ doesn't make use of a_i , we can ignore it, and conclude that $L(i, j) = L(i - 1, j)$. That is, the common subsequence that solves $L(i, j)$ is the same subsequence that solves $L(i - 1, j)$.

Similarly, if $a_i \neq b_j$, and b_j is not part of the solution to $L(i, j)$ but a_i is part of this solution, then $L(i, j) = L(i, j - 1)$.

For the third case, if $a_i \neq b_j$, and neither a_i nor b_j is part of the solution to $L(i, j)$, then $L(i, j) = L(i - 1, j - 1)$ (that is, we can ignore both a_i and b_j).

In general, when $a_i \neq b_j$ we do not know which of the above three cases we are in, so we define our recurrence relation to take the maximum of the values returned by these three cases. That is

$$L(i, j) = \max\{L(i - 1, j), L(i, j - 1), L(i - 1, j - 1)\} \quad \text{if } a_i \neq b_j.$$

Now we can put our two main cases together, to get the whole recurrence relation.

$$L(i, j) = \begin{cases} 1 + L(i - 1, j - 1) & \text{if } a_i = b_j, \\ \max\{L(i - 1, j), L(i, j - 1), L(i - 1, j - 1)\} & \text{if } a_i \neq b_j. \end{cases}$$

This recurrence relation can also be rewritten as

$$L(i, j) = \begin{cases} 1 + L(i - 1, j - 1) & \text{if } a_i = b_j, \\ \max\{L(i - 1, j), L(i, j - 1)\} & \text{if } a_i \neq b_j. \end{cases}$$

The initial conditions are

$$L(i, 0) = 0 \text{ for } 0 \leq i \leq n \quad \text{and} \quad L(0, j) = 0 \text{ for } 0 \leq j \leq m$$

since there can be no common subsequence if one of the sequences is empty.

Notice that we have only provided the maximum *length* of the longest common subsequence. We have not calculated its location in the two arrays, nor have we determined if there might be more than one common subsequence that gives this maximal length.